

컴파일러 과제1

보고서

20181755 이건희

2024년 9월

1. 개발 환경

```
kh@ThinkPad-T16g2: ~  
kh@ThinkPad-T16g2:~$ neofetch  
      ,--/+00SSSS00+/-,  
      `:+SSSSSSSSSSSSSSSS+`  
    -+SSSSSSSSSSSSSSSSyySSSS+-  
    .0SSSSSSSSSSSSSSSSdMMMNySSSS0.  
    /SSSSSSSSSSShdmmNNmyNMMMMhSSSSS/  
    +SSSSSSSSshmydMMMMMMNdddySSSSSSS+  
    /SSSSSSSSshNMMMyhhyyyymNMMMNhSSSSSSS/  
    .SSSSSSSSdMMMNhSSSSSSSSShNMMMdSSSSSSS.  
    +SSShhhyNMMNySSSSSSSSSSyNMMMySSSSSSS+  
    0SSyNMMMNyMMhSSSSSSSSSSShmmhSSSSSSS0  
    0SSyNMMMNyMMhSSSSSSSSSSShmmhSSSSSSS0  
    +SSShhhyNMMNySSSSSSSSSSyNMMMySSSSSSS+  
    .SSSSSSSSdMMMNhSSSSSSSSShNMMMdSSSSSSS.  
    /SSSSSSSShNMMMyhhyyyhdNMMMNhSSSSSSS/  
    +SSSSSSSSdmydMMMMMMNdddySSSSSSS+  
    /SSSSSSSSShdmmNNmyNMMMMhSSSSS/  
    .0SSSSSSSSSSSSSSSSdMMMNySSSS0.  
    -+SSSSSSSSSSSSSSSSyySSSS+-  
      `:+SSSSSSSSSSSSSS+`  
      ,--/+00SSSS00+/-,  
kh@ThinkPad-T16g2  
-----  
OS: Ubuntu 22.04.4 LTS on Windows 10 x86_64  
Kernel: 5.15.153.1-microsoft-standard-WSL2  
Uptime: 3 hours, 13 mins  
Packages: 702 (dpkg), 6 (snap)  
Shell: bash 5.1.16  
Terminal: Windows Terminal  
CPU: 13th Gen Intel i5-1345U (12) @ 2.495GHz  
GPU: 48e0:00:00.0 Microsoft Corporation Device  
Memory: 588MiB / 15827MiB
```

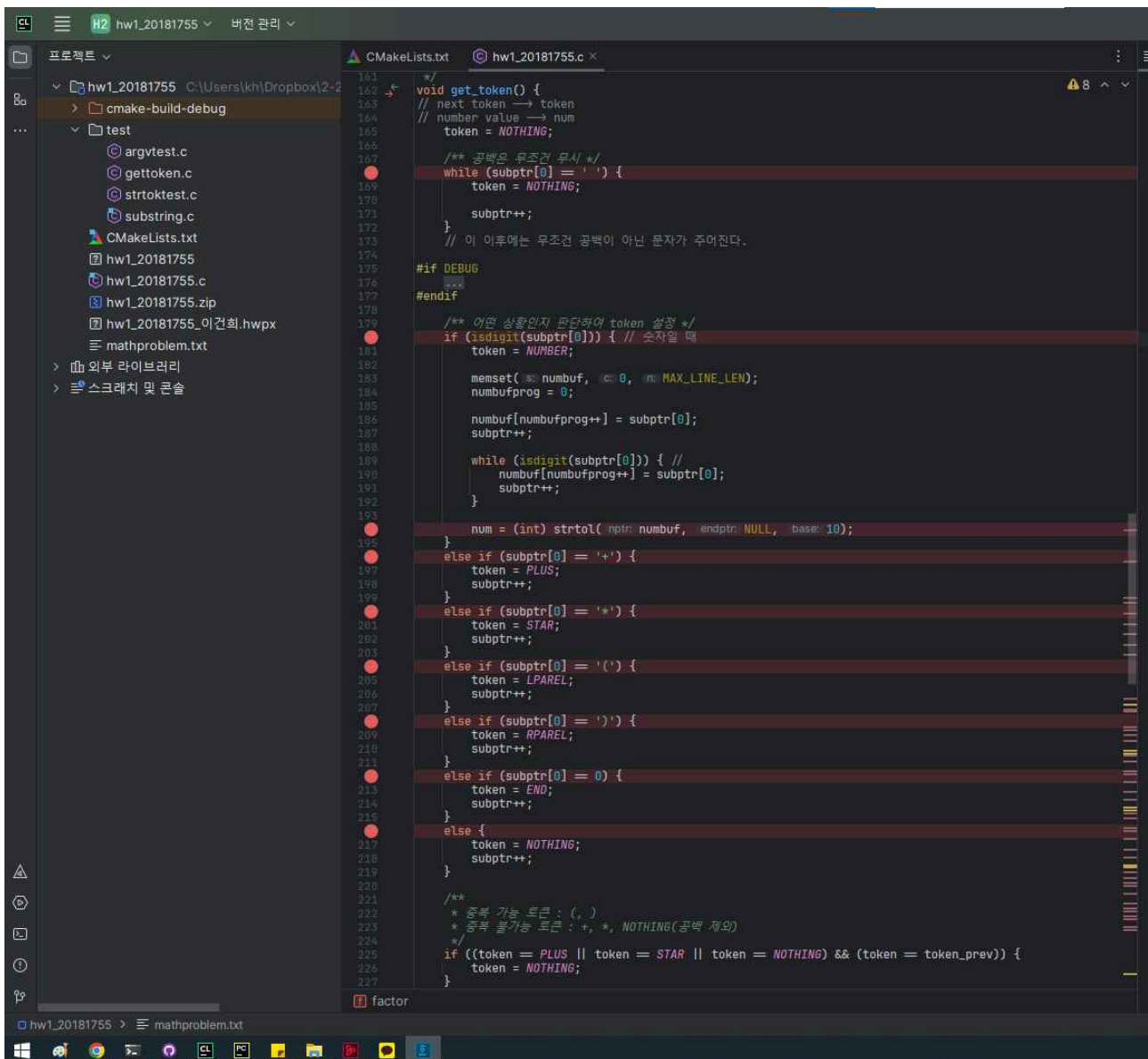
```
kh@ThinkPad-T16g2: ~  
kh@ThinkPad-T16g2:~$ gcc -v  
Using built-in specs.  
COLLECT_GCC=gcc  
COLLECT_LTO_WRAPPER=/usr/lib/gcc/x86_64-linux-gnu/11/lto-wrapper  
OFFLOAD_TARGET_NAMES=nvptx-none:amdgc-nvptx-none:amdhsa  
OFFLOAD_TARGET_DEFAULT=1  
Target: x86_64-linux-gnu  
Configured with: ../src/configure -v --with-pkgversion='Ubuntu 11.4.0-1ubuntu1~22.04' --with-bugurl=file:///usr/share/doc/gcc-11/README.Bugs --enable-languages=c,ada,c++,go,brig,d,fortran,objc,obj-c++,m2 --prefix=/usr --with-gcc-major-version-only --program-suffix=-11 --program-prefix=x86_64-linux-gnu- --enable-shared --enable-linker-build-id --libexecdir=/usr/lib --without-included-gettext --enable-threads=posix --libdir=/usr/lib --enable-nls --enable-bootstrap --enable-clocale=gnu --enable-libstdc++-debug --enable-libstdc++-time=yes --with-default-libstdc++-abi=new --enable-gnu-unique-object --disable-vtable-verify --enable-plugin --enable-default-pie --with-system-zlib --enable-libphobos-checking=release --with-target-system-zlib=auto --enable-objc-gc=auto --enable-multiarch --disable-werror --enable-cet --with-arch=32=i686 --with-abi=m64 --with-multilib-list=m32,m64,mx32 --enable-multilib --with-tune=generic --enable-offload-targets=nvptx-none=/build/gcc-11-XeT9LY/gcc-11-11.4.0/debian/tmp-nvptx/usr,amdgc-nvptx-none:amdhsa=/build/gcc-11-XeT9LY/gcc-11-11.4.0/debian/tmp-gcn/usr --without-cuda-driver --enable-checking=release --build=x86_64-linux-gnu --host=x86_64-linux-gnu --target=x86_64-linux-gnu --with-build-config=bootstrap-lto-lean --enable-link-serialization=2  
Thread model: posix  
Supported LTO compression algorithms: zlib zstd  
gcc version 11.4.0 (Ubuntu 11.4.0-1ubuntu1~22.04)  
kh@ThinkPad-T16g2:~$ gdb -v  
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04.2) 12.1  
Copyright (C) 2022 Free Software Foundation, Inc.  
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>  
This is free software: you are free to change and redistribute it.  
There is NO WARRANTY, to the extent permitted by law.
```

ubuntu 22.04 lts @ wsl2, gcc 11, gdb 12



clion에서 개발, 빌드/디버깅 시 상기한 wsl2의 툴체인 사용

2. 코드 설명



```
161 */
162 void get_token() {
163     // next token → token
164     // number value → num
165     token = NOTHING;
166
167     /** 공백은 무조건 무시 */
168     while (subptr[0] == ' ') {
169         token = NOTHING;
170
171         subptr++;
172     }
173     // 이 이후에는 무조건 공백이 아닌 문자가 주어진다.
174
175 #if DEBUG
176     ***
177 #endif
178
179     /** 어떤 상황인지 판단하여 token 설정 */
180     if (isdigit(subptr[0])) { // 숫자일 때
181         token = NUMBER;
182
183         memset(s, numbuf, 0, MAX_LINE_LEN);
184         numbufprog = 0;
185
186         numbuf[numbufprog++] = subptr[0];
187         subptr++;
188
189         while (isdigit(subptr[0])) { //
190             numbuf[numbufprog++] = subptr[0];
191             subptr++;
192         }
193
194         num = (int) strtol(nptr, numbuf, base: 10);
195     }
196     else if (subptr[0] == '+') {
197         token = PLUS;
198         subptr++;
199     }
200     else if (subptr[0] == '*') {
201         token = STAR;
202         subptr++;
203     }
204     else if (subptr[0] == '(') {
205         token = LPAREL;
206         subptr++;
207     }
208     else if (subptr[0] == ')') {
209         token = RPAREL;
210         subptr++;
211     }
212     else if (subptr[0] == 0) {
213         token = END;
214         subptr++;
215     }
216     else {
217         token = NOTHING;
218         subptr++;
219     }
220
221     /**
222      * 중복 가능 토큰 : (, )
223      * 중복 불가능 토큰 : +, *, NOTHING(공백 제외)
224      */
225     if ((token == PLUS || token == STAR || token == NOTHING) && (token == token_prev)) {
226         token = NOTHING;
227     }
228 }
```

죄송합니다 금요일에 예비군 갔다오니 시간이 없어서 보고서까지 설명을 또 적기가 힘드네요.... 주석 상세히 적어놨으니 읽어주시면 감사하겠습니다 직접 만든거 맞습니다

3. 실행

```
kh@ThinkPad-T16g2: ~/com × + ▾
kh@ThinkPad-T16g2:~/compiler/hw1_20181755$ gcc hw1_20181755.c -o hw1_20181755
kh@ThinkPad-T16g2:~/compiler/hw1_20181755$ ./hw1_20181755 mathproblem1.txt
```

프로그램을 실행시 수학문제 파일의 이름을 인수로 넣어야 실행이 된다.

```
kh@ThinkPad-T16g2: ~/com × + ▾
kh@ThinkPad-T16g2:~/compiler/hw1_20181755$ ./hw1_20181755
Error: Missing "math problems file name"
Usage: ./hw1_20181755 <math problems file name>
kh@ThinkPad-T16g2:~/compiler/hw1_20181755$
```

```
kh@ThinkPad-T16g2: ~/com × + ▾
kh@ThinkPad-T16g2:~/compiler/hw1_20181755$ ./hw1_20181755 notvalidfilename.txt
Error : math problem file does not exist
kh@ThinkPad-T16g2:~/compiler/hw1_20181755$
```

인수를 넣지 않거나 올바르지 않은 파일명을 넣으면 경고하고 exit(1) 한다.

다음과 같은 세 가지의 경우를 생각해 볼 수 있다.

- parser가 지원하는 token에 대하여 올바른 수식
- parser가 지원하는 token에 대하여 올바르지 않은 수식
- parser가 지원하지 않는 token이 들어간 수식 (ex. '-', '/', 'a' 등)

mathproblem.txt 파일은 직접 적은 테스트 파일이다. 확인을 위해 더 많은 예제를 추가하고 싶었으나 시간이 모자라서 더 생각 해내지 못했고, 그 대신 작동이 안 되는 예제를 더 많이 추가하였다. :1-5, :6-15, :16-20에 각각 위의 경우 1, 2, 3에 해당하는 식을 넣었다.

```
kh@ThinkPad-T16g2: ~/com × + v
kh@ThinkPad-T16g2:~/compiler/hw1_20181755$ gcc -o hw1_20181755 hw1_20181755.c
kh@ThinkPad-T16g2:~/compiler/hw1_20181755$ ./hw1_20181755 mathproblem.txt
01. (1+2)*3 = 9
02. (((((7*(5+6)))))) = 77
03. 0+999*0+1 = 1
04. 777*123*666 = 63650286
05. (1+2)*3*(4+5)+(6) = 87
06. 1+++++2 : Error -1
07. 9***10 : Error -1
08. 7* : Error -1
09. 8+ : Error -1
10. +8 : Error -1
11. *7 : Error -1
12. ((3+4)))))) : Error -3
13. (5 : Error -2
14. 6) : Error -3
15. 7 8 : Error -3
16. a : Error -1
17. 1-2 : Error -3
18. 3/4 : Error -3
19. 15x16 : Error -3
20. 9*(9-1) : Error -2
kh@ThinkPad-T16g2:~/compiler/hw1_20181755$
```

실행 결과이다.

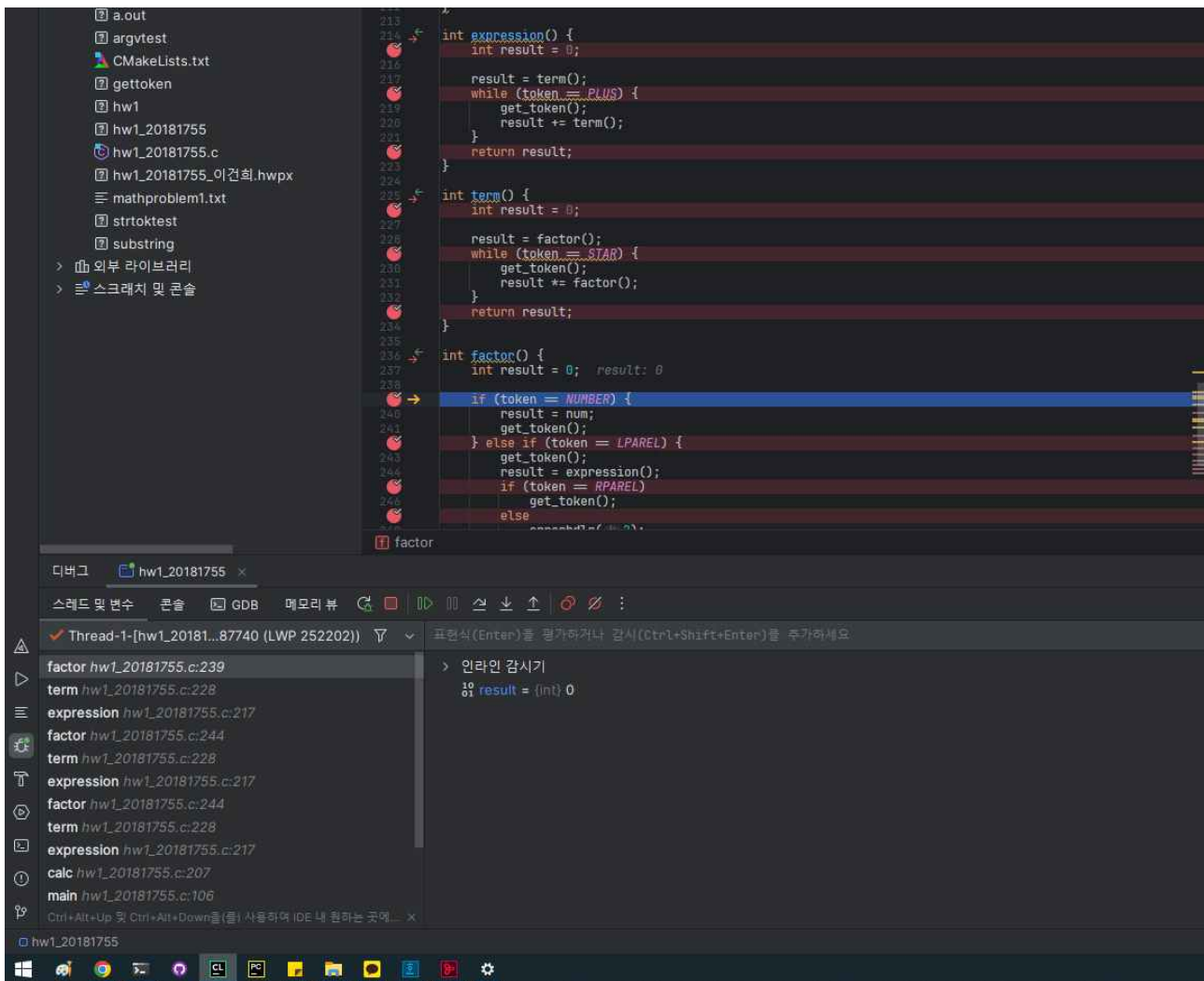
4. 한계

‘-’ (이하 MINUS)을 지원하지 않는다. MINUS는 token의 종류에 없기 때문이다. 뺄셈도 지원하지 않으며 문제의 답이 음수가 나올 수도 없다. 이를 처리하려 한다면 MINUS가 음수를 나타내는 것인지, 수학 연산자 빼기를 나타내는 것인지 구분하는 방법을 생각해 볼 필요가 있다. 또한 ‘/’ (이하 SLASH)를 지원하지 않는다. SLASH 역시 token의 종류에 없기 때문이다. 이를 처리하려 한다면 div by zero를 어떻게 예외처리할지 생각해 볼 필요가 있다. 물론 과제에서는 요구하지 않았으나 추가적으로 수행했다면 좋았을 것이다.

```
327      /** 개선 필요 */
328  → int errorHandler(int i) {
329      switch (i) {
330          case 1:
331              return -1;
332          case 2:
333              return -2;
334          case 3:
335              return -3;
336          default:;
337      }
338  }
```

또한 에러 경우의 수가 주어진 코드에서 3가지가 존재하는데, case 1과 3의 의미를 파악하지 못했다. 각 상황이 어떤 경우인지 알고 이에 맞는 경고를 출력했으면 좋았을 것이다.

5. 결론



디버깅하면서 모든 분기에 breakpoint를 걸고 한 단계씩 진행시키면서 함수가 재귀적으로 실행되어 파싱해 나가는 과정을 확인하였다(위 사진의 왼쪽 아래에서 함수가 재귀적으로 쌓이는 과정을 볼 수 있음). Recursive-Descent 파싱의 과정이 주어져있고, 실제로 완성해야 하는 것은 `get_token` 밖에 없었기에 과제를 해결할 수 있었다.